幸运成像算法的FPGA实现

赵盼孜，李彬华，毛桄晔，陶勇
（昆明理工大学信息工程与自动化学院，云南 昆明 650051）

摘要：幸运成像技术是用于消除天文图像中大气湍流影响的高分辨率图像重建技术，传统的基于CPU的幸运成像算法难于实时化。本文利用FPGA并行处理的优势，设计了一种基于FPGA的幸运成像算法并构建了一个FPGA实验系统。该系统用FPGA完成了选图、配准、叠加的全部幸运成像算法流程，所得高分辨率图像与基于传统CPU算法处理的结果完全相同，但幸运成像算法的处理速度比传统CPU的处理速度快19倍。该算法在FPGA上的实现，为幸运成像技术的实时或准实时化提供了一种可行的方案。

关键词：图像处理；高分辨率图像重建；幸运成像；FPGA

中图分类号：TP391.4 **文献标识码** **文章编号**

引言

地基大口径光学望远镜对天体成像的分辨率受制于大气湍流[1]。为了消除大气湍流对目标成像质量的影响，常见且有效的方法是采用自适应光学（Adaptive Optics，AO）技术，以及事后处理方式的图像复原技术，幸运成像技术就是其中之一。幸运成像技术利用部分短曝光图像中包含的目标高分辨率的结构信息，在观测后进行图像重建，以便消除大气湍流的干扰从而获得高分辨率图像[1, 2]。本世纪初出现的电子倍增电荷耦合器件（Electron Multiplying Charge-Coupled Devices，EMCCD）成像技术具有高分辨率、高读出速度、低噪声等诸多优点，适合于曝光时间在毫秒及其以上量级的微光成像领域[3]。该技术的出现，使得幸运成像技术获得了长足的进步，并在在天文观测领域取得了极大的成功，2013年以前发表的论文就已超过200多篇[2]。典型的幸运成像系统包括英国剑桥大学的LuckyCam、德国马普研究所的AstraLux和西班牙加那利群岛天文研究所的FastCam，他们在双星或多星系统中的暗伴星的发现、天测天力参数计算等方面，科学产出极为丰富[4, 5]。

幸运成像技术是一种简单可行的图像复原方法，但由于图像复原是在观测完成后的一段时间内进行，其缺点也是明显的，即天文观测人员对于所拍摄图像的实时信息了解不多难以及时发现并纠正观测中可能存在的偏差或错误。解决此问题的一个办法是改进算法，增加硬件的处理能力[3]，将幸运成像技术实时或准实时化。

传统的中央处理器（Central Processing Unit，CPU）对图像数据的处理采用的是串行方式，对于大量图像的幸运重建来说，难以实现实时或准实时化。相较于CPU，现场可编程逻辑阵列（Field Programmable Gate Array，FPGA）器件有着先天的并行性和灵活性优势，能提供强大的并行计算能力，可加速数据或信号的处理，特别是图像处理[6]，在天文观测和数据设备中使用越来越多[7]。将FPGA用于幸运成像，是将该技术实时或准实时化的一种有效方法。2008年FastCam项目研究过程中，首次将幸运成像技术在FPGA硬件上实现，并获得了实时的高分辨率图像[1,8]；2015年Jackson也将幸运成像算法在FPGA和GPU中实现[9]。不过，他们在论文中并没有对算法的FPGA实现进行细致的描述。FPGA与幸运成像算法结合的研究工作国内尚无报道。

本文对传统的基于CPU和程序设计语言的幸运成像基本算法进行了简短的分析，结合FPGA技术的特点，提出了一种适合于FPGA处理的幸运成像方案，用Verilog硬件描述语

基金项目：国家自然科学基金（11673009）资助.
收稿日期：
作者简介：赵盼孜,女，硕士研究生. 研究方向：成像技术与图像处理. E-mail：838875923@qq.com
通讯作者：李彬华，男，教授.研究方向：成像技术与图像处理、天文探测器技术. E-mail：lbh@bao.ac.cn

言（HDL）进行各功能模块的设计，并将幸运成像算法成功地在 FPGA 开发板上实现，最后利用中国科学院云南天文台丽江观测站 2.4m 天文望远镜拍摄的大量目标短曝光图像进行硬件算法设计的可行性验证。

1 基于 FPGA 的幸运成像算法处理流程

幸运成像算法的基本思想是按照一定的评价标准，从一系列短曝光图像中选出符合评价标准的“幸运图像”或区域，之后再对其进行配准叠加处理，以获得不受或少受大气湍流影响的高分辨率图像[2]。本文所构建的幸运成像系统在实现过程中按照基本的幸运成像算法的设计要求进行，只是在为了适应本系统所用 FPGA 的处理逻辑与资源限制，在不改变幸运算法基本处理流程（选图、配准、叠加）的基础上将该算法用 Verilog HDL 重新进行了设计。

1.1 幸运图像的选取

在一系列短曝光序列图像中选出“幸运图像”是最终获得高分辨图像的关键因素。然而为获得“幸运图像”必须选取一个合适的像质评价函数。对于空间点源目标图像来说，像质评价函数通常采用图像的斯特列尔比（Strehl Ratio，SR）作为评价标准。由于 SR 定义为存在像差时图像的峰值光强与不存在像差时图像的峰值光强之比，因此，计算时需要理想的无像差峰值光强。但这个理想的峰值光强在现实中并不容易获得，故通常采用其它的替代方案。一个简单且有效的方法是使用瞬时的斯特列尔比作为图像的像质评价函数[2, 10]，即只对图像上的峰值光强像素点的灰度值进行 SR 值估算。

对于 FPGA 这类运算资源有限的硬件，计算 SR 只能采用简单的处理方式。具体来说，在本系统实现过程中，直接使用有像差时图像的峰值光强像素点的灰度值。由于在配准过程中对每帧图像的峰值光强像素点灰度值的位置有要求，故在本系统选图过程中，以图像的峰值光强像素点灰度值及其位置共同对图像进行像质评价。

1.2 图像的配准与叠加

配准是幸运成像处理中最重要的一环，如若配准不当就会直接导致叠加后的图像模糊不清，分辨率下降，从而无法将暗弱目标图像显示出来。由于本系统使用的是恒星的图像，所以最常用的配准方法有两种，一种是以整幅图像中的最大灰值为中心截取所需的成像区域，另一种则是以图像的质心为配准基点。由于 FPGA 硬件逻辑及资源的限制，本系统采用前一种配准方法。

图像配准完成后，则需要对配准图像进行叠加。在本系统中由于所选图像数量偏小，所以采用的直接叠加方法。叠加后所得像图再进行灰度变换（即锐化）便可得到易于观察的高分辨率图像。

2 幸运成像算法的 FPGA 实现

本文所述的幸运成像算法实现的硬件平台是以 Xilinx 公司 Spartan-6 系列的 XC6SLX16 芯片为核心的开发板。开发环境为 ISE Design Suite 14.7，逻辑设计所用 HDL 是 Verilog，采用 ChipScope pro 14.7 和 ModelSim 10.1d 进行硬件设计的验证与调试。

2.1 算法总体设计

由于本项目设计的主要目的是在 FPGA 硬件上实现幸运成像算法，因此为读取图片像素数据，首先将天文短曝光图像存储在开发板外挂的安全数据卡（Secure Digital Card，即 SD 卡）中，以便给算法模块读取图像数据。此外，为满足算法对速度及片内存储的要求，同时由于所用芯片逻辑资源和存储空间的限制，故在数据处理方式上本文采用数据流，即

逐像素处理的方式，但在每个模块内以及各个模块间均采用的是并行的方式对数据进行处理。与此同时在片内数据的存储上为使处理速度和存储器之间达到最佳状态，采用片内的双端口随机存储器（Random Access Memory，RAM）模块作为数据缓存器。然而，由于所用FPGA 芯片 RAM 资源以及逻辑资源的约束，只能对 1000 张 128×128 像素大小的短曝光图像作选图处理，并且在配准过程中只能最大截取 64×64 像素大小的图片进行配准处理，并最终以 64×64 像素大小显示幸运成像结果。

对于整个算法的硬件构架来说，为了算法实现过程中修改及调试的方便，采用了模块化的设计方式。主要模块包括数据的读写、保存模块，短曝光图像的选图、配准、叠加模块以及最终的重建高分辨率图像显示模块，并用 Verilog HDL 进行设计实现。基于 FPGA 实现的幸运成像算法的结构框图如图 1 所示。这一幸运成像的 FPGA 硬件的工作流程可描述如下。

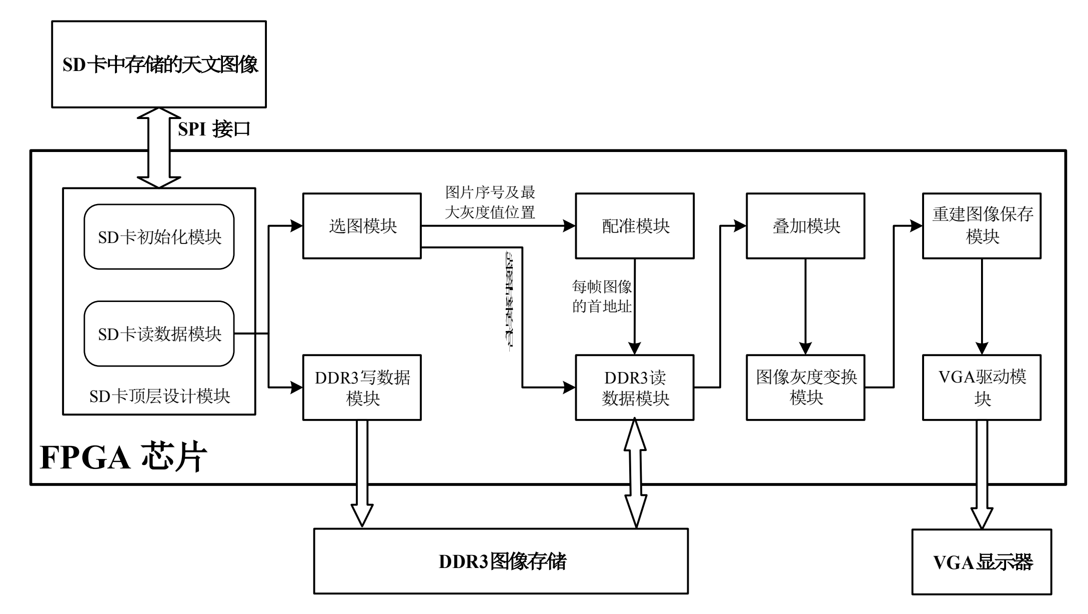


图 1 基于 FPGA 的幸运成像算法的结构框图

Fig. 1 Structure block diagram of lucky imaging algorithm based on FPGA

系统上电后，SD 卡中存储的天文图像数据会通过串行外设接口（Serial Peripheral Interface，SPI）以 6.25Mb/s 的速度不断地向 SD 卡顶层模块传送，经 SD 卡顶层模块处理过的数据经先入先出（First-In First-Out，FIFO）缓冲器不断地向第三代双倍数据率同步动态随机存取存储器（Double-Data-Rate Three Synchronous Dynamic Random Access Memory，DDR3）写数据模块中写入（DDR3 速度 666Mb/s），然后再通过 DDR3 写数据模块的处理，向 FPGA 芯片外部的 DDR3 存储器中写入图像数据。在写数据模块向 DDR3 写入像素数据的同时，选图模块也不断地对每帧图片的像素值进行最大灰度值求解，然后再对求出的所有图片的最大灰度值排序；当排序结束，也就说明选图结束，产生选图结束信号。

其后，便向配准模块发送要截取的原始图片的序号以及该图片最大灰度值位置参数，以便配准模块向 DDR3 读数据模块发送所选图片首像素地址。与此同时，选图结束信号也会发送给 DDR3 读数据模块，当选图结束信号和所选图像首像素地址同时作用于 DDR3 读数据模块时，该模块便会将数据从 DDR3 存储器中读出满足条件的图片像素进行叠加处理。虽然从图 1 的流程上看，配准、DDR3 读出与叠加这三个模块是串行的，但用 HDL 设计及 FPGA 实现时，它们是并行的，它们的运行从时间上看是重叠的。

当所选的全部图片叠加完成后，再将结果进行分段线性灰度变换的处理，以增强目标区域的可视性。然后再将最终高分辨率图像的像素数据保存在片内 RAM 中。当保存完毕后，

再通过视频图形阵列（Video Graphics Array，VGA）驱动模块驱动，将灰度图在 VGA 显示器上显示。

2.2 选图模块的设计

在选图模块的实现上本系统主要采用最大灰度值求解模块以及最大灰度值排序模块共同搭建而成，这两个子模块的搭建主要是由比较器、计数器以及片内 RAM 存储器来完成的。其中最大灰度值查找模块的状态转移图如图 2 所示。

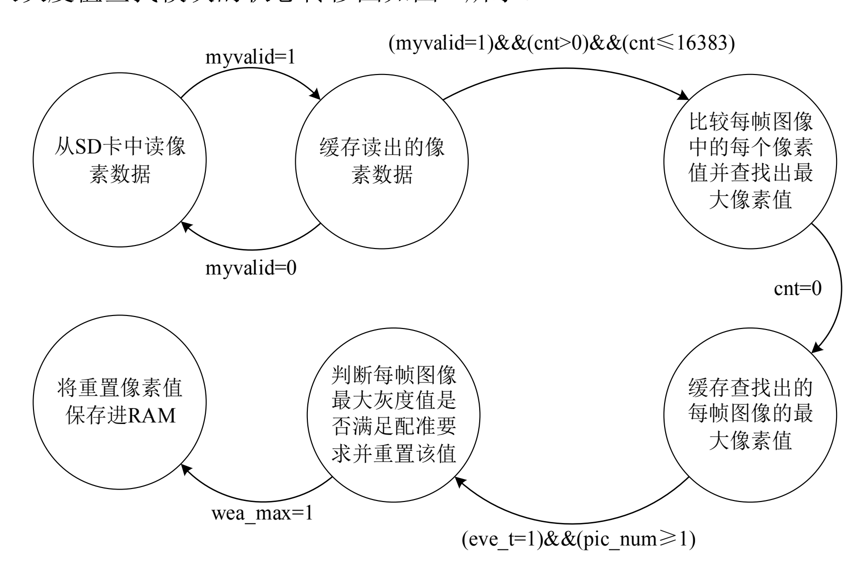


图 2 最大灰度值查找的状态转移图

Fig. 2 State transfer graph of maximum gray value lookup

在图 2 中 myvaild=1 表示 SD 卡中的数据可以取出，cnt 表示已读取的一帧图像的像素数目，其中 cnt=0 表示一帧图像的全部像素已读完。eve_t=1 表示单帧图像的最大灰度值开始取出，pic_num 表示图像的帧数，wea_max=1 表示可以开始向 RAM 中写数据。从 3.1 节中可知，在本设计中所用的图像大小为 128×128，帧数为 1000，所以 cnt 最大为 16383，pic_num 最大为 999。

从图 2 中可以看出，该模块的主要功能是对从 SD 卡中读出的图片像素值逐个进行比较，以找出每帧图像的最大灰度值并将其保存在片内 RAM 中供排序使用。值得注意的是，在每求出一帧图像最大灰度值后，在条件触发时先将其缓存，然后等相应条件到来时再将其进行判断，看其是否满足配准对最大灰度值位置的要求。满足则在 RAM 相应的地址空间中保存该像素值，不满足则在 RAM 相应的地址空间中存入与该最大灰度值相同比特位数的零值。其次，在向 RAM 中保存每帧图像最大灰度值时，要同时将该最大灰度值所在的图片序号及位置参数一并保存在 RAM 相应地址空间中，以便进行并行串行混合排序。

2.3 配准模块的设计

在配准算法的实现上，采用以最大灰度值为图片中心，截取 64×64 像素大小进行叠加处理，其设计主要是通过选图模块中发送的最大灰度值所在图片序号以及其位置参数，再根据式(1)计算出要截取的图片首地址，然后发送给 DDR3 读数据模块，用以从 DRR3 存储器中读出相应的图片像素值，供叠加模块处理，该模块并不占用片内 RAM 资源，只是少量的使用了 Slices 资源。

$$c3_p0_cmd_addr_r = ((pic_cnt-1)\times 65536)+\left(\left(\left(\begin{array}{l}(\mathrm{INT}(\max_cnt/2^7)-31)\times 128\\ +(\max_cnt-\mathrm{INT}(\max_cnt/2^7)\times 128)-31\end{array}\right)-1\right)\times 4\right) \quad (1)$$

其中，$c3_p0_cmd_addr_r$ 表示要读取的 DDR3 的地址，pic_cnt 表示要读取的最大灰度值所在的图片序号，$\max_cnt$ 表示要读取的最大灰度值在图片中的位置参数，$\mathrm{INT}(\max_cnt/2^7)$ 表示 $\max_cnt$ 除以 2^7 之后只取整数部分，在 FPGA 实现时用移位寄存器实现。

2.4 叠加模块的设计

对于图像的叠加主要是通过片内 RAM 资源的使用实现的，在叠加模块设计中占用的片内 RAM 个数为 12 个约 37.5%。该模块的结构框图如图 3 所示。

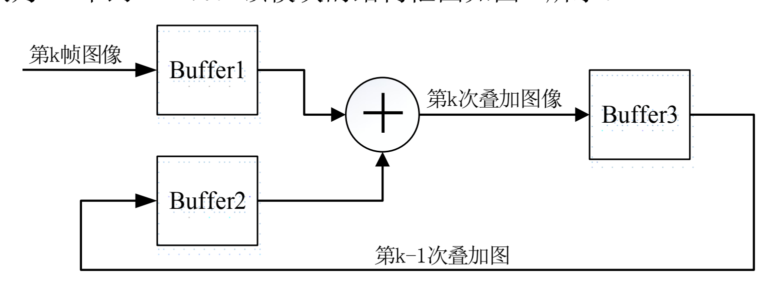


图 3 叠加模块结构框图

Fig. 3 Block diagram of stacking module

3　实验结果及其分析

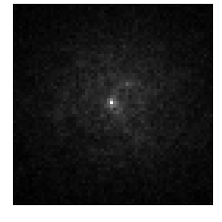
为了验证前面所述的幸运成像算法在 FPGA 上实现的正确性，我们拟用 MATLAB 对相同的幸运成像算法编程并将其在 PC 机上的处理结果与 FPGA 处理结果进行对比。而对于 FPGA 上运行幸运成像算法的处理时间问题，也只能与 PC 机的运行时间做比较。所以本文设计了两个实验，一是与 MATLAB 处理结果的对比实验，一是本系统自身处理结果与数据实验。

幸运成像实验，必须有天文目标短曝光图像。本文采用的是 2016 年 10 月 20 日在云南天文台丽江观测站对天文双星 HDS 70 的观测图像，共 10000 帧，这组图像的观测条件和参数参见毛桃哗等人的论文[10]。在图像帧数以及图像大小的选取上，由于所用的 FPGA 开发板资源的限制，多次随机从 10000 帧图像中抽取 1000 帧，并裁剪成 128×128 像素大小的天文目标短曝光图像进行处理。

3.1 幸运成像算法在 CPU+MATLAB 上的平台上处理实验

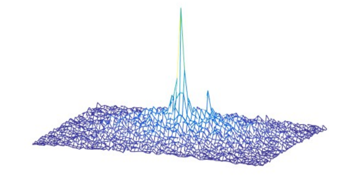
实验硬件平台是：Dell Precision T5500 图像工作站、16GB 内存、NVIDIA GTX1050Ti 显卡，运行的软件环境是 Windows 7 操作系统、MATLAB R2014a。

将随机抽取的 1000 帧 128×128 像素大小的图像进行分组，然后在 MATLAB 分别对每组图像均进行同样的算法处理，在实际处理中我们只截取了 64×64 像素大小的目标区域。根据毛桃哗等人的研究结果[10]，选图比取为 1%，所得到如图 4 所示的高分辨率图像和三维灰度分布图。这一结果与毛桃哗所得结果一致。



(a)　二维灰度图

(a) 2D gray image



(b) 三维灰度分布图

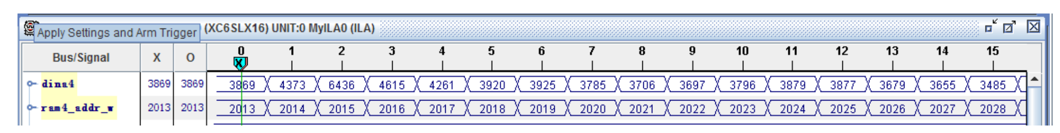
(b) 3D gray distribution map

图 4 1%选图比下所得的高分辨率图像

Fig. 4 High-resolution image obtained at 1% selection rate

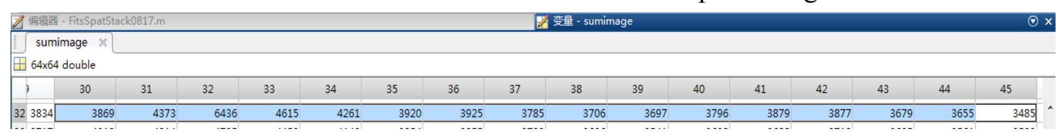
3.2 幸运成像算法在 FPGA 上的实验验证

硬件平台是以 Xilinx 公司 Spartan-6 系列的 XC6SLX16 芯片为核心的开发板。基于 FPGA 上实现的幸运成像算法设计的验证，采用 1%的选图比，即随机的 1000 帧选 10 帧，执行与 MATLAB 相同幸运成像的算法处理，然后用 ChipScope 捕捉最终结果图中最大灰度值周围部分像素数据值，得到了如图 5(a)图所示的经 FPGA 实现的幸运成像算法处理所得的高分辨率图像部分像素数据值，并与如图 5(b)图所示的经 MATLAB 实现的幸运成像算法处理所得的高分辨率图像的相同部分像素值做对比。



(a) FPGA 处理的结果

（a） Results of FPGA processing



(b) MATLAB 处理的结果

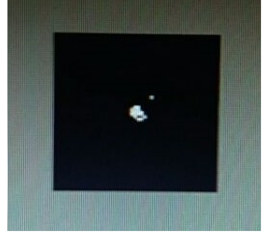
(b) Results of MATLAB processing

图 5 分别用 FPGA 与 MATLAB 处理所得图像像素数值

Fig. 5 Image pixel values processed by FPGA and MATLAB respectively

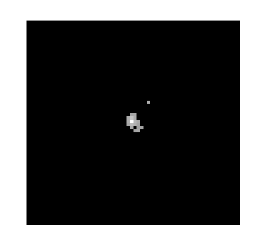
通过图 5 可知，经 FPGA 上实现的幸运成像算法处理所得图像的最终像素数据与在 MATLAB 上做同样算法处理所得图像像素数据完全相同。

从处理效果上来看，最终得到的高分辨率图像像素值较小，直接通过显示器显示图像时，无论在 MATLAB 上还是 FPGA 上做算法处理所得的高分辨率图像都比较模糊，所以，在 FPGA 和 MATLAB 上均对算法处理所得的高分辨率图像做了相同的灰度变换处理——锐化或图像增强，将最终得到的高分辨率图像中感兴趣的目标或区域突出出来，分别得到了如图 6(a)图、6(b)图所示的最终高分辨率图像。



（a）FPGA 处理的结果

(a) Results of FPGA processing



（b）MATLAB 处理的结果

(b) Results of MATLAB processing

图 6 基于 FPGA 和 MATLAB 实现的幸运成像算法处理所得的高分辨率图像

Fig. 6 High-resolution images Obtained by lucky imaging algorithm based on the FPGA and the MATLAB

图 6(a)是经 FPGA 上实现的幸运成像算法处理所得的高分辨率图像以 1024×768 的分辨率在 XGA 显示器上显示并通过手机拍摄得到的，在此采用手机拍摄获得，是因为本系统实现所采用的 FPGA 开发板是一个独立的外部系统，它是脱离 PC 机单独工作的，所以基于 FPGA 硬件实现的幸运成像算法处理后所得的高分辨率图像要通过单独的 XGA 显示器进行显示，因此它与 PC 机上 MATLAB 所得高分辨率结果图获取方式不同，它不能像

MATLAB 一样直接进行图像截屏，而只能采用外部的图像采集设备进行图像获取，所以经 FPGA 上实现的幸运成像算法处理所得的高分辨率图像在本文中采用手机拍摄得到。而图 6(b)经 MATLAB 处理后显示的最终高分辨率图像是以 1280×768 的 WXGA 分辨率显示的。由于两图的显示器分辨率不同导致纵横比不同，导致相同图像在显示上稍有差异。此外，图 6(a)是手机拍摄获得的，由于手机拍摄角度及分辨率的不同所以导致图像有些许变化。不过，仍然可看出这两幅图像是基本相同的，这说明经 FPGA 上实现的幸运成像算法处理后的结果与经 MATLAB 实现的幸运成像算法处理后的结果一致，从而说明基于 FPGA 实现的幸运成像算法设计正确。

综上所述，无论是从最终高分辨率图像中的像素数据值对比来看，还是从灰度变换后的直观显示出来最终高分辨率图像对比来看，都说明了基于 FPGA 实现的幸运成像算法设计与实现的正确性。

3.3 FPGA 上实现幸运成像算法的优势

FPGA 的并行性和灵活性是其最大的优势，因此相较于传统的 CPU，FPGA 对同样算法的处理速度肯定快。但对于所研究的幸运成像算法来说，具体快多少，必须通过实验来说明。为此，分别在 FPGA 和 CPU+MATLAB 上对相同幸运算法的运算速度进行了测试。实验过程中，统计 1000 帧图像的处理时间，选图比采用 1%。FPGA 及 CPU+MATLAB 平台上均采用 2.1 节中介绍的算法进行实现。由此得到了不同平台下相同算法的平均运行时间：对于 FPGA 是 2.45s，对于 CPU+MATLAB 是 46.93s；运行所得结果(高分辨率图像)分别如图 6(a)、6(b)图所示。显然，相同的算法在不同的平台处理下，得到相同的高分辨率图像，FPGA 平台上算法处理速度比 CPU+MATLAB 平台上算法处理速度快约 19 倍。如果采用更先进的 FPGA 器件，速度还将加快。

不过受制于 SD 卡的速度，总体运行时间并没有明显的优势。目前，我们正在设计基于 USB3 和千兆以太网的高速数据接口，以加速整个幸运成像处理的速度。另外，采用处理能力更强、逻辑资源更多的 FPGA 器件，可以更多地引入并行计算、减少串行处理过程，充分发挥 FPGA 硬件加速的优势。

4　结论

本文首先简述了幸运算法的基本思想及处理流程，根据所用 FPGA 的硬件资源，用 Verilog HDL 设计了幸运成像算法的选图、配准、叠加等关键模块，详细介绍了这些模块的实现方案，并在 FPGA 开发板上实现了这一幸运成像系统。然后，通过将 FPGA 处理的结果与 PC 上用 MATLAB 处理得到的结果进行对比，证明了此系统设计和实现过程的正确性。最后，通过对 FPGA 与 CPU+MATLAB 平台下实现相同算法所需的处理时间的对比分析，说明了 FPGA 在处理速度上快 19 倍的优势。虽然本设计方案还有一些可以改进之处，但就幸运成像算法在 FPGA 开发板上的具体实现，为构建实时化的高速幸运成像系统探索了一种可行的技术方法。

致谢

中国科学院云南天文台张西亮、季凯帆、金振宇为本研究工作提供了原始天文图像，特此致谢。

参考文献:

[1] Oscoz A., Rebolo R., Lopez R., et al. FastCam: a new lucky imaging instrument for medium-sized telescopes, Proc. Of SPIE, 2008, 7014: 701447.

[2] Mackay C.D., High-Efficiency Lucky Imaging[J], MNRAS, 2013, 432(1), 702-710.
[3] 胡泊, 李彬华. 低温下EMCCD电子倍增模型[J]. 电子学报, 2013, 41(9):1826-1830.
Hu B., Li B.. Electron Multiplication Model of EMCCD in Low Temperature[J]. Acta Electronica Sinica, 2013, 41(9):1826-1830.
[4] Law N.M., Hodgkin S.T., Mackay C.D.. The LuckyCam survey for very low mass binaries - II. 13 new M4.5-M6.0 binaries, MNRAS, 2008, 384(1): 150-160.
[5] Woller M., Bandner W., A Lucky Imaging search for stellar sources near 74 transit hosts, 2015, A&A, 579, A129.
[6] Treece B. CPU or FPGA for image processing: Which is best? [J], Vision Systems Design, 2017, 22(8): 23-24.
[7] 印琪骏, 朱亮, 等. 基于高性能芯片的射电天文厘米-分米波谱线观测方法[J]. 天文研究与技术, 2017, 14(1): 103-109.
Yin Q., Zhu L., et al. A New Method to Observe Radio Astronomy Signal: Centimeter-Decimeter Spectral Line——Based on High Performance Integrated Circuit. Astronomical Research & Technology, 2017, 14(1): 103-109.
[8] Piqueras J, Rodriguez-Ramos L, Martin Y, et al. FastCam: Real-Time Implementation of the Lucky Imaging Technique using FPGA[C]. Programmable Logic, 2008, Southern Conference on. IEEE, 2008:155-160.
[9] Jackson C R. Real time mitigation of atmospheric turbulence in long distance imaging using the lucky region fusion algorithm with FPGA and GPU hardware acceleration[D]. University of Delaware, 2015.
[10] 毛栊哗，李彬华，张西亮，季凯帆，金振宇. 基于2m级大口径望远镜的幸运成像算法的实验研究[J]. 光学技术，2018，44(5)（待刊）.
Mao L., Li B., Zhang X., Ji K., Jin Zh.. Experimental investigation of lucky imaging algorithm based on 2.4m astronomical telescope [J]. Opt Techn, 2018, 44(5)（In press）.

Implementation of Lucky Imaging Algorithm Based on FPGA

Zhao Panzi, Li Binhua, Mao Longhau, Tao Yong

(Faculty of information engineering and automation, Kunming University of Science and Technology, Kunming, Yunnan 650051, China)

ABSTRACT: Lucky imaging is a kind of high-resolution image reconstruction techniques used to eliminate the influence of atmospheric turbulence on astronomical images. Its algorithm is usually implemented on the CPU-based platform, and cannot meet the requirement of real-time high-resolution imaging. Taking advantages of FPGA parallel processing, this paper presents a new lucky imaging algorithm based on FPGA, and builds an experimental system which uses FPGA to complete all the lucky imaging processes of the image selection, registration, stacking. The obtained image reconstructed by the FPGA system is exactly the same with the image processed by the traditional CPU platform. Nevertheless, it is important that the processing speed of the system is 19 times faster than that of the CPU platform. The implementation of the algorithm on FPGA provides us with a feasible scheme to make lucky imaging technology real-time or quasi real-time.

Keywords: Image processing; High-resolution image reconstruction; Lucky imaging; FPGA